**LAB # 09**

**Task 1: You are working on a project where you get data from third party webapi / service in XML format. You are creating charts from that data for creating a chart you need data in JSON format in order to make them compatible you need to convert xml data into json but limitation is you can’t modify third party webservice / api you have to make Adapter that returns data in JSON by using adapter pattern.**

**Solution :**

**Abstract Class Processor**

internal abstract class Processor

{

protected Processor successor;

public void SetSuccessor(Processor successor)

{

this.successor = successor;

}

public abstract void ProcessRequest(Processes purchase);

}

**Class Processes**

internal class Processes

{

private int number;

private string name;

public Processes(int num, string name)

{

this.number = num;

this.name = name;

}

public int Number

{

get { return number; }

set { number = value; }

}

public string Name

{

get { return name; }

set { name = value; }

}

}

**Class Media Player**

internal class MediaPlayer : Processor

{

public override void ProcessRequest(Processes processes)

{

if (processes.Name.ToLower() == "media")

Console.WriteLine("\n{0} processed the process #{1} : {2}", this.GetType().Name, processes.Number, processes.Name);

else if (successor != null)

successor.ProcessRequest(processes);

}

}

**Class Notepad**

internal class Notepad : Processor

{

public override void ProcessRequest(Processes processes)

{

if (processes.Name.ToLower() == "document")

Console.WriteLine("\n{0} processed the process #{1} : {2}", this.GetType().Name, processes.Number, processes.Name);

else if (successor != null)

successor.ProcessRequest(processes);

}

}

**Class Browser**

internal class Browser : Processor

{

public override void ProcessRequest(Processes processes)

{

if (processes.Name.ToLower() == "internet")

Console.WriteLine("\n{0} processed the process #{1} : {2}", this.GetType().Name, processes.Number, processes.Name);

else

Console.WriteLine("\nProcess #{0} : {1}, cannot be processed", processes.Number, processes.Name);

}

}

**Class Calculator**

internal class Calculator : Processor

{

public override void ProcessRequest(Processes processes)

{

if (processes.Name.ToLower() == "calculation")

Console.WriteLine("\n{0} processed the process #{1} : {2}", this.GetType().Name, processes.Number, processes.Name);

else if (successor != null)

successor.ProcessRequest(processes);

}

}

**Main Method**

static void Main(string[] args)

{

Processor calculator = new Calculator();

Processor mediaplayer = new MediaPlayer();

Processor notepad = new Notepad();

Processor browser = new Browser();

calculator.SetSuccessor(mediaplayer);

mediaplayer.SetSuccessor(notepad);

notepad.SetSuccessor(browser);

Processes pd = new Processes(3056, "media");

calculator.ProcessRequest(pd);

pd = new Processes(3057, "internet");

calculator.ProcessRequest(pd);

pd = new Processes(3058, "calculation");

calculator.ProcessRequest(pd);

pd = new Processes(3059, "document");

calculator.ProcessRequest(pd);

pd = new Processes(3060, "game");

calculator.ProcessRequest(pd);

}

**OUTPUT :**

